Івашко В.Д.
науковий керівник к.т.н. доцент Бочкарьов О. Ю.

Розподілена комп'ютерна система для розв'язку задачі про призначення

Розподілені комп'ютерні системи для розв'язку **задачі про призначення** (також відомої як задача призначень або Assignment Problem) використовуються для ефективного розподілу ресурсів, завдань або робіт між агентами (наприклад, машинами, людьми, чи процесами). Задача про призначення є класичною оптимізаційною задачею [1], де потрібно знайти оптимальний спосіб призначення завдань виконавцям, мінімізуючи витрати або максимізуючи вигоду.

Основні аспекти побудови розподіленої системи для розв'язання задачі про призначення:

1. Модель задачі - заснована на двочастковому графі, де одна частина відповідає завданням, а інша — агентам. Ребра з'єднують завдання та агентів із вартістю або вигодою, пов'язаною з їх призначенням. Ціль — мінімізувати загальні витрати або максимізувати загальну вигоду від призначення завдань агентам.[2]
2. Для централізованих рішень використовується угорський алгоритм або його варіації (наприклад, для задач мінімізації або максимізації). У розподіленій системі можна використовувати варіанти угорського алгоритму із паралельними або розподіленими обчисленнями, де кожен вузол системи відповідає за частину графу або за певний піднабір завдань [3].
3. Розподілена архітектура - до прикладу модель\ клієнт-сервер - завдання призначень може вирішуватись на сервері, де всі дані збираються й обробляються централізовано. Розподілена модель: кожен вузол системи (процесор або машина) може обробляти певну частину завдань і взаємодіяти з іншими вузлами для обміну результатами або частковими рішеннями. Це дозволяє масштабувати обчислення для великих задач. Мережеві протоколи: для обміну даними між вузлами можуть використовуватися протоколи типу MPI (Message Passing Interface), gRPC або HTTP API.

Огляд сучасних підходів і мотивація для розподілених систем:

Зростаюча складність і об'єм даних в задачах призначення вимагають переходу до розподілених систем. Традиційні централізовані рішення не справляються з масштабуванням і можуть бути неефективними у випадку великих наборів даних або складних задач. Тому в рамках даної магістерської роботи прийнято рішення

використовувати клієнт-серверну архітектуру на базі Docker, що розподілена система складається з двох основних компонентів:

Клієнт – це програма або вузол, який відправляє запити до сервера для виконання певних завдань, наприклад, для вирішення задачі про призначення. Клієнт може бути веб-додатком, мобільним додатком або іншим інтерфейсом користувача.

Сервер – це центральний компонент системи, який приймає запити від клієнтів, виконує необхідні обчислення або обробляє запити і повертає результати назад клієнтам. Сервев виконує основну частину логіки, пов'язану із розв'язуванням задачі про призначення, складається з кількох контейнерів, які паралельно обробляють окремі задачі.

На початкових етапах угорського алгоритму можна розподілити обчислення мінімумів для кожного рядка і стовпця між різними контейнерами. Наприклад, один контейнер обробляє перші 100 рядків матриці, інший — наступні 100 рядків, і так далі. Те саме відбувається для стовпців. Процес пошуку оптимальних "покриттів" нулями (перевірка, чи схема системи можливо покрити всі нулі мінімальною кількістю ліній) також може бути розподілений. Різні контейнери можуть аналізувати різні частини матриці для визначення мінімального покриття. Для цього контейнеризація дозволяє легко розділити великий граф на підграфи, кожен з яких може оброблятися окремим контейнером.

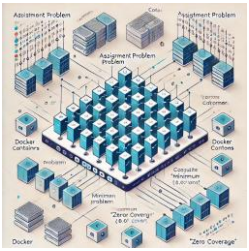

Рис.1 Схема системи

Після того як знайдені покриття для нулів, контейнери можуть паралельно виконувати призначення завдань на основі мінімальних значень у відповідних підграфах.У випадку, якщо у вас є багато виконавців і завдань, можна використовувати паралельні процеси для обробки різних підгруп, щоб зменшити час обчислень.

Література

1. "Distributed Systems: Principles and Paradigms" – Ендрю С. Таненбаум, Маартен ван Стеен
2. "The Design and Analysis of Algorithms" – Анупам Кумар.
3. "Approximation Algorithms" – Віджай В. Вазірані